Imperial College
Cluster2 FGM

User's Manual Appendix I
Parameterised Command Sequences

Document Ref. :- User's Manual \ Appndx I PCS
Issue : - 3
Rev. : - 0
Date : - 13th July 2000

Revisions

Issue	Rev.	Date	Sec.	Change
Draft	0.2	10/03/95		FGMOPM5 corrected to mode 0x3 from 0x5
Draft	0.2	10/03/95		OPM5 corrected to mode 0xD from 0x4
Draft	0.2	13/03/95	All	Added ESOC commands
Draft	0.2	13/03/95		Added calibration mode definition
1	0	27/11/95	All	General revision all sections. ZEF1 commands added to Class 3.
1	0	27/11/95	3.4	**set_b2_offset** 'send ZEF2PBAS, 0x13'
1	0	27/11/95	3.4	**set_bl2_offset** 'send ZEF2PBAS, 0x16'
1	0	27/11/95	3.4	**set_zy2_threshold** 'send ZEF2PBAS, 0x18'
1	0	27/11/95	2.2	**FGMOPM1_to_FGMOPM4** 'send ZEF2ENTN, 0x1'
1	0	27/11/95	2.2	**FGMOPM1_to_FGMOPM5** 'send ZEF2ENTN, 0x1'
2	0	13/5/99	All	All sections revised for Cluster2 Document reference changed to: 'Users Manual \ Appndx I PCS'
3	0	3/7/00		Section 5 JSOC/ESOC command sequences aliases added

Contents

1. Introduction

The FGM instruments have a large set of defined telecommands. The actions of each is set out in the Software Requirements Document, which was written to control the development of the embedded software. These commands form a low level instruction set for the instruments. However, writing control sequences in this 'machine code' is difficult, so blocks of commands have been built up to form 'commanding functions'. The document has been written to define these 'commanding functions' and to explain their use.

The effect a telecommand has, depends on the state of the instrument when it is received. In the most extreme case, with the instrument in its off state, all telecommands are ignored. To describe each command rigorously, some assumptions must be made about the state of an instrument. For this reason the 'commanding functions' have been grouped into three classes. These are defined as follows:

'Class 1' Commanding functions which change the operating mode of the instrument. The operating modes are those set out in chapter 6 of the Instrument User's Manual. The full effect of these functions on the housekeeping is defined.

'Class 2' Commanding functions for routine use, which do not change the operating mode of the instrument. The full effect of these functions on the housekeeping is defined.

'Class 3' Commanding functions which can be used only when the FGM is in 'Engineering Mode'. The effect of a commanding function on the housekeeping may depend on other Class 3 commands which have been sent previously.

If it is assumed that an instrument is switched on and booted using the 'default boot sequence' (Instrument User's Manual, Appendix C), then Class 1 and Class 2 command functions can be sent in any order. Provided that an instrument remains 'on', the effect of a function on the housekeeping is clearly defined.

1.1 Use of Parameters.

When FGM Command Sequences take parameters, the type of the parameter is often a 16-bit un-signed integer. Where so, in order to accommodate the JSOC/ESOC commanding scenario, the FGM sequence is also given with an equivalent 8-bit parameter list.

Parameter type used in FGM Command Sequence

BYTE	8 bit unsigned integer
WORD	16 bit unsigned integer
INT16	16 bit signed integer

Operators used in FGM Command sequences

&	bit-wise AND
|	bit-wise OR
^	bit-wise EXOR
>>	shift right
<<	shift left

2. Class 1 Commanding Functions

2.1 Description of Class 1

By carefully defining a small number of software configurations, and fixing the hardware configuration in which they can be used, an FGM instrument can be viewed as a simple state machine. The set of commands in Class 1 execute transitions around the associated state diagram.

Use of FGMOPM7 and FGMOPM8 is tied to the use of BM1 and BM3 telemetry acquisition by the spacecraft. If an instrument is commanded to provide data at a higher rate than the spacecraft is acquiring, some data will be lost. The instrument will function correctly but disruption of the data stream should be avoided when possible. To prevent this occurring the, spacecraft should be switched to a higher data rate before the instrument is switched, and the instrument should be switched to a lower data rate before the spacecraft is switched.

2.2 Class 1: Operating mode changes

off_to_FGMOPM1

off_to_FGMOPM1()
Parameters
void
Parameter range limits
N/A
Action
This function is implemented by the 'default instrument boot sequence' listed in appendix C of the FGM Instrument User's Manual. It takes the instrument from its 'off' state to operating mode FGMOPM1.
HK parameters changed
On completion of the boot sequence, the housekeeping parameters will be those shown in Table 2-1 of the Instrument User's Manual.
FGM Sequence

```
off_to_FGMOPM1()
{
/* default instrument boot sequence */
}
```

FGMOPM1_to_off

FGMOPM1_to_off()
Parameters
void
Parameter range limits
N/A
Action
This is the normal powering off sequence for an FGM instrument.
HK parameters changed
All FGM housekeeping parameters become invalid.
FGM Sequence

```
FGMOPM1_off()
{
/* spacecraft to switch off primary power to instrument */
}
```

FGMOPM1_to_FGMOPM2

FGMOPM1_to_FGMOPM2()

Parameters

void

Parameter range limits

N/A

Action

Change the operating mode of the instrument from FGMOPM1 to FGMOPM2 (see FGM Instrument User's Manual for details of operating modes).

HK parameters changed

EF3TCREC -> EF3TCREC + 1
EF3TMOPT = 0xC -> EF3TMOPT = 0xB

FGM Sequence

```
FGMOPM1_to_FGMOPM2()
{
  send ZEF2TMMS,0xB
}
```

FGMOPM1_to_FGMOPM3

FGMOPM1_to_FGMOPM3()

Parameters

void

Parameter range limits

N/A

Action

Change the operating mode of the instrument from FGMOPM1 to FGMOPM3 (see FGM Instrument User's Manual for details of operating modes).

HK parameters changed

EF3TCREC -> EF3TCREC + 1
EF3TMOPT = 0xC -> EF3TMOPT = 0xA

FGM Sequence

```
FGMOPM1_to_FGMOPM3()
{
  send ZEF2TMMS,0xA
}
```

FGMOPM1_to_FGMOPM4

FGMOPM1_to_FGMOPM4()

Parameters

void

Parameter range limits

N/A

Action

Change the operating mode of the instrument from FGMOPM1 to FGMOPM4 (see FGM Instrument User's Manual for details of operating modes).

HK parameters changed

EF3TCREC -> EF3TCREC + 3
EF3TMOPT = 0xC -> EF3TMOPT = 0x4

FGM Sequence

```
FGMOPM1_to_FGMOPM4()
{
  send ZEF2ENTN,0x1
  send ZEF2TRGS,0x1
  send ZEF2TMMS,0x4
}
```

FGMOPM1_to_FGMOPM5

FGMOPM1_to_FGMOPM5()

Parameters

void

Parameter range limits

N/A

Action

Change the operating mode of the instrument from FGMOPM1 to FGMOPM5 (see FGM Instrument User's Manual for details of operating modes).

HK parameters changed

EF3TCREC -> EF3TCREC + 3

EF3TMOPT = 0xC -> EF3TMOPT = 0x3

FGM Sequence

```
FGMOPM1_to_FGMOPM5()
{
  send ZEF2ENTN,0x1
  send ZEF2TRGS,0x1
  send ZEF2TMMS,0x3
}
```

FGMOPM1_to_FGMOPM7

FGMOPM1_to_FGMOPM7()

Parameters

void

Parameter range limits

N/A

Action

Change the operating mode of the instrument from FGMOPM1 to FGMOPM7 (see FGM Instrument User's Manual for details of operating modes).

Note :

To avoid a loss of data, this function should be executed after the s/c telemetry acquisition is changed from NM1,NM2, NM3 or BM2 to BM1.

HK parameters changed

EF3TCREC -> EF3TCREC + 1

EF3TMOPT = 0xC -> EF3TMOPT = 0xD

FGM Sequence

```
FGMOPM1_to_FGMOPM7()
{
  send ZEF2TMMS,0xD
}
```

FGMOPM1_to_FGMOPM8

FGMOPM1_to_FGMOPM8()

Parameters

void

Parameter range limits

N/A

Action

Change the operating mode of the instrument from FGMOPM1 to FGMOPM8 (see FGM Instrument User's Manual for details of operating modes).

Note :

To avoid a loss of data, this function should be executed after the s/c telemetry acquisition is changed from NM1,NM2, NM3 or BM2 to BM3.

HK parameters changed

EF3TCREC -> EF3TCREC + 1

EF3TMOPT = 0xC -> EF3TMOPT = 0xF

FGM Sequence

```
FGMOPM1_to_FGMOPM8()
{
  send ZEF2TMMS,0xF
}
```

FGMOPM2_to_FGMOPM1

FGMOPM2_to_FGMOPM1()

Parameters

void

Parameter range limits

N/A

Action

Change the operating mode of the instrument from FGMOPM2 to FGMOPM1 (see FGM Instrument User's Manual for details of operating modes).

HK parameters changed

EF3TCREC -> EF3TCREC + 1
EF3TMOPT = 0xB -> EF3TMOPT = 0xC

FGM Sequence

```
FGMOPM2_to_FGMOPM1()
{
  send ZEF2TMMS,0xC
}
```

FGMOPM2_to_FGMOPM3

FGMOPM2_to_FGMOPM3()

Parameters

void

Parameter range limits

N/A

Action

Change the operating mode of the instrument from FGMOPM2 to FGMOPM3 (see FGM Instrument User's Manual for details of operating modes).

HK parameters changed

EF3TCREC -> EF3TCREC + 1
EF3TMOPT = 0xB -> EF3TMOPT = 0xA

FGM Sequence

```
FGMOPM2_to_FGMOPM3()
{
  send ZEF2TMMS,0xA
}
```

FGMOPM3_to_FGMOPM1

FGMOPM3_to_FGMOPM1()

Parameters

void

Parameter range limits

N/A

Action

Change the operating mode of the instrument from FGMOPM3 to FGMOPM1 (see FGM Instrument User's Manual for details of operating modes).

HK parameters changed

EF3TCREC -> EF3TCREC + 1
EF3TMOPT = 0xA -> EF3TMOPT = 0xC

FGM Sequence

```
FGMOPM3_to_FGMOPM1()
{
  send ZEF2TMMS,0xC
}
```

FGMOPM3_to_FGMOPM2

FGMOPM3_to_FGMOPM2()

Parameters

void

Parameter range limits

N/A

Action

Change the operating mode of the instrument from FGMOPM3 to FGMOPM2 (see FGM Instrument User's Manual for details of operating modes).

HK parameters changed

EF3TCREC -> EF3TCREC + 1
EF3TMOPT = 0xA -> EF3TMOPT = 0xB

FGM Sequence

```
FGMOPM3_to_FGMOPM2()
{
  send ZEF2TMMS,0xB
}
```

FGMOPM4_to_FGMOPM1

FGMOPM4_to_FGMOPM1()

Parameters

void

Parameter range limits

N/A

Action

Change the operating mode of the instrument from FGMOPM4 to FGMOPM1 (see FGM Instrument User's Manual for details of operating modes).

HK parameters changed

EF3TCREC -> EF3TCREC + 1
EF3TMOPT = 0x4 -> EF3TMOPT = 0xC

FGM Sequence

```
FGMOPM4_to_FGMOPM1()
{
  send ZEF2TMMS,0xC
}
```

FGMOPM5_to_FGMOPM1

FGMOPM5_to_FGMOPM1()

Parameters

void

Parameter range limits

N/A

Action

Change the operating mode of the instrument from FGMOPM5 to FGMOPM1 (see FGM Instrument User's Manual for details of operating modes).

HK parameters changed

EF3TCREC -> EF3TCREC + 1
EF3TMOPT = 0x3 -> EF3TMOPT = 0xC

FGM Sequence

```
FGMOPM5_to_FGMOPM1()
{
  send ZEF2TMMS,0xC
}
```

FGMOPM6_to_FGMOPM1

FGMOPM6_to_FGMOPM1()

Parameters

void

Parameter range limits

N/A

Action

Change the operating mode of the instrument from FGMOPM6 to FGMOPM1 (see FGM Instrument User's Manual for details of operating modes).

HK parameters changed

EF3TCREC -> EF3TCREC + 1

EF3TMOPT = 0x2 -> EF3TMOPT = 0xC

FGM Sequence

```
FGMOPM6_to_FGMOPM1()
{
  send ZEF2TMMS,0xC
}
```

FGMOPM7_to_FGMOPM1

FGMOPM7_to_FGMOPM1()

Parameters

void

Parameter range limits

N/A

Action

Change the operating mode of the instrument from FGMOPM7 to FGMOPM1 (see FGM Instrument User's Manual for details of operating modes).

Note :

To avoid a loss of data, this function should be executed before the s/c telemetry acquisition is changed from BM1 to a lower FGM data rate.

HK parameters changed

EF3TCREC -> EF3TCREC + 1

EF3TMOPT = 0xD -> EF3TMOPT = 0xC

FGM Sequence

```
FGMOPM7_to_FGMOPM1()
{
  send ZEF2TMMS,0xC
}
```

FGMOPM8_to_FGMOPM1

FGMOPM8_to_FGMOPM1()

Parameters

void

Parameter range limits

N/A

Action

Change the operating mode of the instrument from FGMOPM8 to FGMOPM1 (see FGM Instrument User's Manual for details of operating modes).

Note :

To avoid a loss of data, this function should be executed before the s/c telemetry acquisition is changed from BM3 to a lower FGM data rate.

HK parameters changed

EF3TCREC -> EF3TCREC + 1

EF3TMOPT = 0xF -> EF3TMOPT = 0xC

FGM Sequence

```
FGMOPM8_to_FGMOPM1()
{
  send ZEF2TMMS,0xC
}
```

FGMOPM1_to_FGMEXT

FGMOPM1_to_FGMEXT()

Parameters

void

Parameter range limits

N/A

Action

Change the operating mode of the instrument from FGMOPM1 to FGMEXT (see FGM Instrument User's Manual for details of operating modes).

HK parameters changed

Not applicable – no HK data generated during mode FGMEXT

FGM Sequence

```
FGMOPM1_to_FGMEXT()
{
  zef2srns,0x0007
  zef2pbas,0x0063
  zef2pbys,0x00ff
}
```

FGMEXT_to_FGMOPM1

FGMEXT_to_FGMOPM1()

Parameters

void

Parameter range limits

N/A

Action

Change the operating mode of the instrument from FGMEXT to FGMOPM1 (see FGM Instrument User's Manual for details of operating modes).

HK parameters changed

Not applicable – no HK data generated during mode FGMEXT

FGM Sequence

```
FGMEXT_to_FGMOPM1()
{
  zef2trgs,0x0000
  zef2srns,0x0000
  zef2tmms,0xc
}
```

FGMCAL

FGMCAL(rng, mode)

Parameters

BYTE rng range
BYTE mode calibration mode

Parameter range limits

0 <= mode <= 3
2 <= rng <= 7

Action

This function executes a sequence of commands which facilitate in-flight calibration of an FGM instrument. In terms of mode changes, the instrument switches from FGMOPM1 to FGMCAL and executes a sequence of instructions, then switches from FGMCAL to FGMOPM1 on completion (see FGM Instrument User's Manual for details of operating modes). For the duration of FGMCAL mode, data in the science telemetry stream and on the Inter-Experiment Link is not valid for normal science purposes.

Note :
Receivers of IEL data should be notified by the Project when FGMCAL is active. Instrument is left with primary sensor outboard, secondary sensor inboard and both sensors auto-ranging.

HK parameters changed

EF3TCREC -> EF3TCREC + 20 on exit
EF3CALMD -> 'mode' for a number of resets which depends on the telemetry mode
EF3OCALN -> undefined
EF3PRISR -> 1 on exit
EF3SECSR -> 0 on exit
EF3PARNE -> 1 on exit
EF3SARNE -> 1 on exit

FGM Sequence

```
FGMCAL(rng , mode)
{
  send ZEF2PRNs,0x7
  send ZEF2SRNS,0x7
  send ZEF2POBS,0x1
  send ZEF2SOBS,0x0
  send ZEF2PRNS,(rng)
  send ZEF2SRNS,0x0
  send ZEF2CLBS,(mode)
wait 200 seconds
  send ZEF2PRNs,0x7
  send ZEF2SRNS,0x7
  send ZEF2POBS,0x0
  send ZEF2SOBS,0x1
  send ZEF2PRNS,(rng)
  send ZEF2SRNS,0x0
  send ZEF2CLBS,(mode)
wait 200 seconds
  send ZEF2PRNs,0x7
  send ZEF2SRNS,0x7
  send ZEF2POBS,0x1
  send ZEF2SOBS,0x0
  send ZEF2PRNS,0x0
  send ZEF2SRNS,0x0
}
```

FGMOPM1_to_FGMENG

FGMOPM1_to_ENG()

Parameters

void

Parameter range limits

N/A

Action

Change the operating mode of the instrument from FGMOPM1 to FGMENG ie switch to engineering mode.

Note :

Receivers of IEL data should be notified by the Project when FGMENG is active.

HK parameters changed

None

FGM Sequence

```
FGMOPM1_to ENG()
{
  /* No commands are required to execute the change to engineering mode.
  However, instruments receiving data on the IEL should be notified when the
  engineering mode is active. */
}
```

FGMENG_to_FGMOPM1

ENG_to_FGMOPM1()

Parameters

void

Parameter range limits

N/A

Action

Change the operating mode of the instrument from FGMENG to FGMOPM1, ie switch out of engineering mode. If a fault is diagnosed it may be necessary to reconfigure the instrument while in the engineering mode. In such a case the definitions of all the operating modes will need to be updated to take account of the change.

Note :

On exiting from FGMENG a review must be made of this state of the instrument to establish the extent to which its configuration differs from that reached by following the default instrument boot sequence. Based on this information the 'HK parameters changed' section must be checked for all Class 1 and Class 2 commands

HK parameters changed

The definition of the expected housekeeping parameters should be reviewed on exiting from FGMENG.

FGM Sequence

```
FGMENG_to_FGMOPM1()
{
  /* No commands are required to execute the change from engineering mode. */
}
```

3. Class 2 Commanding Functions

3.1 Description of Class 2

Commands in this class are used for control of aspects of the instrument which change neither the operating mode of the instrument nor its hardware configuration. These restriction mean that housekeeping parameters changed by each commanding function can be defined.

3.2 Class 2: Sensor heater controls

enable_sensor_heaters

enable_sensor_heaters()
Parameters
void
Parameter range limits
N/A
Action
Command spacecraft to switch on FGM sensor heaters.
HK parameters changed
See s/c housekeeping.
FGM Sequence

```
enable_sensor_heaters()
{
/*spacecraft command to turn on FGM sensor heaters */
}
```

disable_sensor_heaters

disable_sensor_heaters()
Parameters
void
Parameter range limits
N/A
Action
Command spacecraft to switch off FGM sensor heaters.
HK parameters changed
See s/c housekeeping.
FGM Sequence

```
disable_sensor_heaters()
{
/*spacecraft command to turn off FGM sensor heaters */
}
```

3.3 Class 2: Direct controls for MSA actions

enable_event_recognition

enable_event_recognition()
Parameters
void
Parameter range limits
N/A
Action
Set the event recognition on.
HK parameters changed
EF3TCREC -> EF3TCREC + 1
EF3EVENT = 1

FGM Sequence

```
enable_event_recognition()
{
  send ZEF2ENTN,1
}
```

disable_event_recognition

disable_event_recognition()
Parameters
void
Parameter range limits
N/A
Action
Set the event recognition off.
HK parameters changed
EF3TCREC -> EF3TCREC + 1
EF3EVENT = 0

FGM Sequence

```
disable_event_recognition()
{
  send ZEF2ENTN,0
}
```

set_msa_trigger

set_msa_trigger()

Parameters

void

Parameter range limits

N/A

Action

Trigger the MSA.

HK parameters changed

EF3TCREC -> EF3TCREC + 1

EF3TRIGG = 1 (FOR ONE TELEMETRY FORMAT ONLY)

FGM Sequence

```
set_msa_trigger()
{
  send ZEF2TRGS,0
}
```

set_msa_untrigger

set_msa_untrigger()

Parameters

void

Parameter range limits

N/A

Action

Un-trigger the MSA.

HK parameters changed

EF3TCREC -> EF3TCREC + 1

EF3TRIGG -> 1 (for one reset period only)

FGM Sequence

```
set_msa_untrigger()
{
  send ZEF2TRGS,1
}
```

3.4 Class 2: Parameter table entries

dump_parameter_table

dump_parameter_table()

Parameters

void

Parameter range limits

N/A

Action

Dump the contents of the parameter table to the science telemetry stream.

HK parameters changed

EF3TCREC -> EF3TCREC + 6
EF3PBUPD = 1 (FOR ONE TELEMETRY RESET PERIOD ONLY)
EF3PBREC = 4
EF3MDUMP = 1 (FOR ONE TELEMETRY RESET PERIOD ONLY)

FGM Sequence

```
dump_parameter_table()
{
  send ZEF2PBAS,1
  send ZEF2PBYS,0x00
  send ZEF2PBYS,0x1e
  send ZEF2PBYS,0x4f
  send ZEF2PBYS,0x00
  send ZEF2DMPS
}
```

set_keyhole_address

set_keyhole_address(addr)

Parameters

WORD addr address

Parameter range limits

0x0000 <= addr <= 0xffff

Action

Set the address of a word, in the processor address space, the contents of which will appear as the 'keyhole' word in the housekeeping.

HK parameters changed

EF3TCREC -> EF3TCREC + 3
EF3PBUPD = 1 (FOR ONE TELEMETRY RESET ONLY)
EF3PBREC = 2
EF3KEYHO MAY CHANGE

FGM Sequence

```
set_keyhole_address(addr)
{
  send ZEF2PBAS 0x0
  send ZEF2PBYS (addr & Ox00ff)
  send ZEF2PBYS (addr & 0xff00) >> 8
}
```

Equivalent sequence with 8-bit parameter list:

```
(a_lsb , a_msb)
{
  send ZEF2PBAS 0x0
  send ZEF2PBYS (a_lsb)
  send ZEF2PBYS (a_msb)
}
```

set_memory_dump_params

set_memory_dump_params(base_addr, len)

Parameters

WORD	base_addr	base address
WORD	len	length

Parameter range limits

0x0000 <= base_address <= 0xffff
0 <= len <= 0x0100

Action

Set the parameters used by the memory dump command to fill the science telemetry stream with contents of RAM. Base address is the address of the first word to be transmitted. Length is the number of words to be transmitted.

HK parameters changed

EF3TCREC -> EF3TCREC + 5
EF3PBUPD = 1 (FOR ONE TELEMETRY PERIOD ONLY)
EF3PBREC = 4

FGM Sequence

```
set_memory_dump_params(base_addr, len)
{
  send ZEF2PBAS 1
  send ZEF2PBYS (base_addr & 0x00ff)
  send ZEF2PBYS (base_addr & 0xff00) >>8
  send ZEF2PBYS (len & 0x00ff)
  send ZEF2PBYS (len & 0xff00) >>8
}

Equivalent sequence with 8-bit parameter list:
(a_lsb , a_msb , l_lsb , l_msb)
{
  send ZEF2PBAS 1
  send ZEF2PBYS (a_lsb)
  send ZEF2PBYS (a_msb)
  send ZEF2PBYS (l_lsb)
  send ZEF2PBYS (l_msb)
}
```

set_averaging_lengths

```
set_averaging_lengths(short, long)
Parameters
WORD      short         short average length
WORD      long          long average length

Parameter range limits
0 < short <= 0x7FFF
0 < long <= 0x7FFF

Action
Set the lengths of the short and long averages which are used by the event
detection.

HK parameters changed
EF3TCREC -> EF3TCREC + 5
EF3PBUPD = 1 (FOR ONE TELEMETRY PERIOD ONLY)
EF3PBREC = 4

FGM Sequence
set_averaging_lengths (short, long)
{
  send ZEF2PBAS,0x3
  send ZEF2PBYS,(short & 0x00ff)
  send ZEF2PBYS,(short & 0xff00) >>8
  send ZEF2PBYS,(long & 0x00ff)
  send ZEF2PBYS (long & 0xff00) >>8
}

Equivalent sequence with 8-bit parameter list:
(s_lsb , s_msb , l_lsb , l_msb)
{
  send ZEF2PBAS,0x3
  send ZEF2PBYS,(s_lsb)
  send ZEF2PBYS,(s_msb)
  send ZEF2PBYS,(l_lsb)
  send ZEF2PBYS (l_msb)
}
```

set_variance_threshold

```
set_variance_threshold(rng, thold)
Parameters
BYTE      rng           range
INT16     thold         threshold

Parameter range limits
0 <= rng <=7
0x8000 <= thold <=0x7fff

Action
Set the variance threshold used by the event detection for a particular range.

HK parameters changed
EF3TCREC -> EF3TCREC + 3
EF3PBUPD = 1 (FOR ONE TELEMETRY PERIOD ONLY)
EF3PBREC = 2

FGM Sequence
set_variance_threshold(rng, thold)
{
  send ZEF2PBAS,(0x5 + rng)
  send ZEF2PBYS,(thold & 0x00ff)
  send ZEF2PBYS,(thold & 0xff00) >> 8
}

Equivalent sequence with 8-bit parameter list:
(rng_plus_5, t_lsb , t_msb)
{
  send ZEF2PBAS,(rng_plus_5)
  send ZEF2PBYS,(t_lsb)
  send ZEF2PBYS,(t_msb)
}
```

set_variance_offset

```
set_variance_offset(oset_msw, oset_lsw)
Parameters
WORD      oset_msw      offset high word
WORD      oset_lsw      offset low word

Parameter range limits
0x0 <= oset_msw <= 0xffff
0x0 <= oset_lsw <= 0xffff

Action
Set the 32 bit variance offset used by the event detection.

HK parameters changed
EF3TCREC -> EF3TCREC + 5
EF3PBUPD = 1 (FOR ONE TELEMETRY PERIOD ONLY)
EF3PBREC = 4

FGM Sequence
set_variance_offset (oset_msw, oset_lsw)
{
  send ZEF2PBAS,0xd
  send ZEF2PBYS,(oset_msw & Ox00ff)
  send ZEF2PBYS,(oset_msw & 0xff00) >> 8
  send ZEF2PBYS,(oset_lsw & Ox00ff)
  send ZEF2PBYS,(oset_lsw & 0xff00) >> 8
}

Equivalent sequence with 8-bit parameter list:
(omsw_lsb , omsw_msb , olsw_lsb , olsw_msb)
{
  send ZEF2PBAS,0xd
  send ZEF2PBYS,(omsw_lsb)
  send ZEF2PBYS,(omsw_msb)
  send ZEF2PBYS,(olsw_lsb)
  send ZEF2PBYS,(olsw_msb)
}
```

set_bx_threshold

```
set_bx_threshold (thold)
Parameters
WORD    thold          threshold

Parameter range limits
0x0 <= thold <= 0xffff

Action
Set the 16 bit threshold for the parameter Bx used by the event detection

HK parameters changed
EF3TCREC -> EF3TCREC + 3
EF3PBUPD = 1 (FOR ONE TELEMETRY PERIOD ONLY)
EF3PBREC = 2

FGM Sequence
set_bx_threshold(thold)
{
  send ZEF2PBAS,0xf
  send ZEF2PBYS,(thold & Ox00ff)
  send ZEF2PBYS,(thold & 0xff00) >> 8
}

Equivalent sequence with 8-bit parameter list:
(t_lsb , t_msb)
{
  send ZEF2PBAS,0xf
  send ZEF2PBYS,(t_lsb)
  send ZEF2PBYS,(t_msb)
}
```

set_bx_offset

set_bx_offset(oset_msw, oset_lsw)

Parameters

```
WORD    oset_msw       offset high word
WORD    oset_lsw       offset low word
```

Parameter range limits

```
0x0 <= oset_msw <= 0xffff
0x0 <= oset_lsw <= 0xffff
```

Action

Set the 32 bit offset for the Bx parameter used by the event detection.

HK parameters changed

```
EF3TCREC -> EF3TCREC + 5
EF3PBUPD = 1 (FOR ONE TELEMETRY PERIOD ONLY)
EF3PBREC = 4
```

FGM Sequence

```
set_bx_offset(oset_msw, oset_lsw)
{
  send ZEF2PBAS,0x10
  send ZEF2PBYS,(oset_msw & Ox00ff)
  send ZEF2PBYS,(oset_msw & 0xff00) >> 8
  send ZEF2PBYS,(oset_lsw & Ox00ff)
  send ZEF2PBYS,(oset_lsw & 0xff00) >> 8
}

Equivalent sequence with 8-bit parameter list:
(omsw_lsb , omsw_msb , olsw_lsb , olsw_lsb)
{
  send ZEF2PBAS,0x10
  send ZEF2PBYS,(omsw_lsb)
  send ZEF2PBYS,(omsw_msb)
  send ZEF2PBYS,(olsw_lsb)
  send ZEF2PBYS,(olsw_msb)
}
```

set_b2_threshold

set_b2_threshold (thold)

Parameters

```
WORD    thold          threshold
```

Parameter range limits

```
0x0 <= thold <= 0xffff
```

Action

Set the 16 bit threshold for the parameter B squared used by the event detection

HK parameters changed

```
EF3TCREC -> EF3TCREC + 3
EF3PBUPD = 1 (FOR ONE TELEMETRY PERIOD ONLY)
EF3PBREC =2
```

FGM Sequence

```
set_b2_threshold(thold)
{
  send ZEF2PBAS,0x12
  send ZEF2PBYS,(thold & Ox00ff)
  send ZEF2PBYS,(thold & 0xff00) >> 8
}

Equivalent sequence with 8-bit parameter list:
(t_lsb , t_msb)
{
  send ZEF2PBAS,0x12
  send ZEF2PBYS,(t_lsb)
  send ZEF2PBYS,(t_msb)
}
```

set_b2_offset

set_b2_offset(oset_msw, oset_lsw)

Parameters

WORD	oset_msw	offset high word
WORD	oset_lsw	offset low word

Parameter range limits

0x0 <= oset_msw <= 0xffff
0x0 <= oset_lsw <= 0xffff

Action

Set the 32 bit offset for the B squared parameter used by the event detection.

HK parameters changed

EF3TCREC -> EF3TCREC + 5
EF3PBUPD = 1 (FOR ONE TELEMETRY PERIOD ONLY)
EF3PBREC = 4

FGM Sequence

```
set_b2_offset(oset_msw, oset_lsw)
{
  send ZEF2PBAS,0x13
  send ZEF2PBYS,(oset_msw & Ox00ff)
  send ZEF2PBYS,(oset_msw & 0xff00) >> 8
  send ZEF2PBYS,(oset_lsw & Ox00ff)
  send ZEF2PBYS,(oset_lsw & 0xff00) >> 8
}

Equivalent sequence with 8-bit parameter list:
(omsw_lsb , omsw_msb , olsw_lsb , olsw_lsb)
{
  send ZEF2PBAS,0x13
  send ZEF2PBYS,(omsw_lsb)
  send ZEF2PBYS,(omsw_msb)
  send ZEF2PBYS,(olsw_lsb)
  send ZEF2PBYS,(olsw_msb)
}
```

set_bl2_threshold

set_bl2_threshold (thold)

Parameters

WORD	thold	threshold

Parameter range limits

0x0 <= thold <= 0xffff

Action

Set the 16 bit threshold for the parameter BL squared used by the event detection

HK parameters changed

EF3TCREC -> EF3TCREC + 3
EF3PBUPD = 1 (FOR ONE TELEMETRY PERIOD ONLY)
EF3PBREC = 2

FGM Sequence

```
set_bl2_threshold(thold)
{
  send ZEF2PBAS,0x15
  send ZEF2PBYS,(thold & Ox00ff)
  send ZEF2PBYS,(thold & 0xff00) >> 8
}

Equivalent sequence with 8-bit parameter list:
(t_lsb , t_msb)
{
  send ZEF2PBAS,0x15
  send ZEF2PBYS,(t_lsb)
  send ZEF2PBYS,(t_msb)
}
```

set_bl2_offset

```
set_bl2_offset(oset_msw, oset_lsw)
```

Parameters

```
WORD    oset_msw       offset high word
WORD    oset_lsw       offset low word
```

Parameter range limits

```
0x0 <= oset_msw <= 0xffff
0x0 <= oset_lsw <= 0xffff
```

Action

Set the 32 bit offset for the BL squared parameter used by the event detection.

HK parameters changed

```
EF3TCREC -> EF3TCREC + 5
EF3PBUPD = 1 (FOR ONE TELEMETRY PERIOD ONLY)
EF3PBREC = 4
```

FGM Sequence

```
set_bl2_offset(oset_msw, oset_lsw)
{
  send ZEF2PBAS,0x16
  send ZEF2PBYS,(oset_msw & Ox00ff)
  send ZEF2PBYS,(oset_msw & 0xff00) >> 8
  send ZEF2PBYS,(oset_lsw & Ox00ff)
  send ZEF2PBYS,(oset_lsw & 0xff00) >> 8
}

Equivalent sequence with 8-bit parameter list:
(omsw_lsb , omsw_msb , olsw_lsb , olsw_lsb)
{
  send ZEF2PBAS,0x16
  send ZEF2PBYS,(omsw_lsb)
  send ZEF2PBYS,(omsw_msb)
  send ZEF2PBYS,(olsw_lsb)
  send ZEF2PBYS,(olsw_msb)
}
```

set_zy2_threshold

```
set_zy2_threshold (thold)
```

Parameters

```
WORD    thold          threshold
```

Parameter range limits

```
0x0 <= thold <= 0xffff
```

Action

Set the 16 bit threshold for the parameter ZY squared used by the event detection

HK parameters changed

```
EF3TCREC -> EF3TCREC + 3
EF3PBUPD = 1 (FOR ONE TELEMETRY PERIOD ONLY)
EF3PBREC = 2
```

FGM Sequence

```
set_zy2_threshold(thold)
{
  send ZEF2PBAS,0x18
  send ZEF2PBYS,(thold & Ox00ff)
  send ZEF2PBYS,(thold & 0xff00) >> 8
}

Equivalent sequence with 8-bit parameter list:
(t_lsb , t_msb)
{
  send ZEF2PBAS,0x18
  send ZEF2PBYS,(t_lsb)
  send ZEF2PBYS,(t_msb)
}
```

set_zy2_offset

```
set_zy2_offset(oset_msw, oset_lsw)
```

Parameters

```
WORD    oset_msw       offset high word
WORD    oset_lsw       offset low word
```

Parameter range limits

```
0x0 <= oset_msw <= 0xffff
0x0 <= oset_lsw <= 0xffff
```

Action

Set the 32 bit offset for the ZY squared parameter used by the event detection.

HK parameters changed

```
EF3TCREC -> EF3TCREC + 5
EF3PBUPD = 1 (FOR ONE TELEMETRY RESET PERIOD ONLY)
EF3PBREC = 4
```

FGM Sequence

```
set_zy2_offset(oset_msw, oset_lsw)
{
  send ZEF2PBAS,0x19
  send ZEF2PBYS,(oset_msw & Ox00ff)
  send ZEF2PBYS,(oset_msw & 0xff00) >> 8
  send ZEF2PBYS,(oset_lsw & Ox00ff)
  send ZEF2PBYS,(oset_lsw & 0xff00) >> 8
}

Equivalent sequence with 8-bit parameter list:
(omsw_lsb , omsw_msb , olsw_lsb , olsw_msb)
{
  send ZEF2PBAS,0x19
  send ZEF2PBYS,(omsw_lsb)
  send ZEF2PBYS,(omsw_msb)
  send ZEF2PBYS,(olsw_lsb)
  send ZEF2PBYS,(olsw_msb)
}
```

set_detection_switches

```
set_detection_switches(edand, edor)
Parameters
WORD    edand          EDAND word
WORD    edor           EDOR word

Parameter range limits
0 <= edand <= 0x001f
0 <= edor <= 0x001f

Action
Set the EDAND and EDOR detection switch words.

HK parameters changed
EF3TCREC -> EF3TCREC + 5
EF3PBUPD = 1 (FOR ONE TELEMETRY PERIOD ONLY)
EF3PBREC = 4

FGM Sequence
set_detection_switches(edand, edor)
{
  send ZEF2PBAS,0x1c
  send ZEF2PBYS,(edand & Ox00ff)
  send ZEF2PBYS,(edadn & 0xff00) >> 8
  send ZEF2PBYS,(edor & Ox00ff)
  send ZEF2PBYS,(edor & 0xff00) >> 8
}

Equivalent sequence with 8-bit parameter list:
(a_lsb , a_msb , o_lsb , o_msb)
{
  send ZEF2PBAS,0x1c
  send ZEF2PBYS,(a_lsb)
  send ZEF2PBYS,(a_msb)
  send ZEF2PBYS,(o_lsb)
  send ZEF2PBYS,(o_msb)
}
```

set_modeswitch_time

```
set_modeswitch_time(count)
Parameters
WORD    count          count of reset periods

Parameter range limits
0 <= count <= 0x7fff

Action
Set the number of resets between an event trigger and a mode switch (if
applicable).

HK parameters changed
EF3TCREC -> EF3TCREC + 3
EF3PBUPD = 1 (FOR ONE TELEMETRY PERIOD ONLY)
EF3PBREC = 2

FGM Sequence
set_modeswitch_time(count)
{
  send ZEF2PBAS,0x1e
  send ZEF2PBYS,(count & Ox00ff)
  send ZEF2PBYS,(count & 0xff00) >> 8
}

Equivalent sequence with 8-bit parameter list:
(c_lsb , c_msb)
{
  send ZEF2PBAS,0x1e
  send ZEF2PBYS,(c_lsb)
  send ZEF2PBYS,(c_msb)
}
```

set_ib_offsets

```
set_ib_offsets(rng, xoffset, yoffset, zoffset)
```

Parameters

BYTE	rng	range
WORD	xoffset	X offset
WORD	yoffset	Y offset
WORD	zoffset	Z offset

Parameter range limits

```
0 <= rng <= 7
0x0 <= xoffset <= 0xffff
0x0 <= yoffset <= 0xffff
0x0 <= zoffset <= 0xffff
```

Action

Set the offsets for the three axes in the given range for the inboard magnetometer.

HK parameters changed

```
EF3TCREC -> EF3TCREC + 7
EF3PBUPD = 1 (FOR ONE TELEMETRY PERIOD ONLY)
EF3PBREC = 6
```

FGM Sequence

```
set_ib_offsets(rng, xoffset, yoffset, zoffset)
{
  send ZEF2PBAS,(0x1f + (rng *3))
  send ZEF2PBYS,(xoffset & Ox00ff)
  send ZEF2PBYS,(xoffset & 0xff00) >> 8
  send ZEF2PBYS,(yoffset & Ox00ff)
  send ZEF2PBYS,(yoffset & 0xff00) >> 8
  send ZEF2PBYS,(zoffset & Ox00ff)
  send ZEF2PBYS,(zoffset & 0xff00) >> 8
}

Equivalent sequence with 8-bit parameter list:
(rng_times_3_plus_0x1f , x_lsb , x_msb , y_lsb , y_msb , z_lsb , z_msb)
{
  send ZEF2PBAS,(rng_times_3_plus_0x1f)
  send ZEF2PBYS,(x_lsb)
  send ZEF2PBYS,(x_msb)
  send ZEF2PBYS,(y_lsb)
  send ZEF2PBYS,(y_msb)
  send ZEF2PBYS,(z_lsb)
  send ZEF2PBYS,(z_msb)
}
```

set_ob_offsets

```
set_ob_offsets(rng, xoffset, yoffset, zoffset)
```

Parameters

```
BYTE    rng            range
WORD    xoffset        X offset
WORD    yoffset        Y offset
WORD    zoffset        Z offset
```

Parameter range limits

```
0 <= rng <= 7
0x0 <= xoffset <= 0xffff
0x0 <= yoffset <= 0xffff
0x0 <= zoffset <= 0xffff
```

Action

Set the offsets for the three axes in the given range for the outboard magnetometer.

HK parameters changed

```
EF3TCREC -> EF3TCREC + 7
EF3PBUPD = 1 (FOR ONE TELEMETRY PERIOD ONLY)
EF3PBREC = 6
```

FGM Sequence

```
set_ob_offsets(rng, xoffset, yoffset, zoffset)
{
  send ZEF2PBAS,(0x37 + (rng *3))
  send ZEF2PBYS,(xoffset & Ox00ff)
  send ZEF2PBYS,(xoffset & 0xff00) >> 8
  send ZEF2PBYS,(yoffset & Ox00ff)
  send ZEF2PBYS,(yoffset & 0xff00) >> 8
  send ZEF2PBYS,(zoffset & Ox00ff)
  send ZEF2PBYS,(zoffset & 0xff00) >> 8
}

Equivalent sequence with 8-bit parameter list:
(rng_times_3_plus_0x37 , x_lsb , x_msb , y_lsb , y_msb , z_lsb , z_msb)
{
  send ZEF2PBAS,(rng_times_3_plus_0x37)
  send ZEF2PBYS,(x_lsb)
  send ZEF2PBYS,(x_msb)
  send ZEF2PBYS,(y_lsb)
  send ZEF2PBYS,(y_msb)
  send ZEF2PBYS,(z_lsb)
  send ZEF2PBYS,(z_msb)
}
```

3.5 Class 2: Others

set_primary_range

```
set_primary_range(rng)
Parameters
BYTE    rng             range

Parameter range limits
0 <= rng <= 7

Action
Set the primary sensor range.

HK parameters changed
EF3TCREC -> EF3TCREC + 1
IF (rng > 0)
EF3PRIMR = rng
EF3PARNE = 0
END
IF (rng = 0)
2 <= EF3PRIMR <= 7
EF3PARNE = 1
END
FGM Sequence
set_primary_range(rng)
{
  send ZEF2PRNS,(rng)
}
```

set_secondary_range

```
set_secondary_range(rng)
Parameters
BYTE    rng             range

Parameter range limits
0 <= rng <= 7

Action
Set the secondary sensor range.

HK parameters changed
EF3TCREC -> EF3TCREC + 1
IF (rng > 0)
EF3SECOR = rng
EF3SARNE = 0
END
IF (rng = 0)
2<= EF3SECOR <= 7
EF3SARNE = 1
END
FGM Sequence
set_secondary_range(rng)
{
  send ZEF2SRNS,(rng)
}
```

4. Class 3 Commanding Functions

4.1 Description of Class 3

Class 3 command functions are provided for use in Engineering Mode. This mode is used to diagnose a suspected malfunction and to reconfigure an instrument accordingly. The effect of a function on the housekeeping can depend on other Class 3 functions which have been sent previously. For this reason the effects of each function on the housekeeping are not listed.

If an instrument is reconfigured in Engineering Mode the definitions of Class 1 and Class 2 commanding functions in this document may no longer be valid. It is necessary for all of these definitions to be checked and updated on any occasion when Class 3 commanding functions have been sent.

4.2 Class 3: Engineering actions

dump_register_zero

```
dump_register_zero()
Parameters
void
Parameter range limits
N/A
Action
Dump the contents of register 0, if the s/c interface ASIC to the housekeeping
stream.
FGM Sequence
dump_register_zero()
{
  send ZEF1DR0S
}
```

send_data_word

```
send_data_word(data)
Parameters
WORD           data           16 bit data word
Parameter range limits
0x0 <= data <= 0xffff
Action
Send a general purpose 16 bit data word to ML2 stack.
This command is intended mainly for use with the code patch command and should
not be in normal use. It could be used to send any required bit sequence,
replacing any other ML2 command.
FGM Sequence
send_data_word (data)
{
  send ZEF2MLTS,(data)
}

Equivalent sequence with 8-bit parameter list:
(data_lsb, data_msb)
{
  WORD data
  data=(data_msb << 8) | data_lsb
  send ZEF2MLTS,(data)
}
```

send_ml1_word

send_ml1_word(data)

Parameters

WORD data 16 bit data word

Parameter range limits

0x0 <= data <= 0xffff

Action

Send a general purpose 16 bit data word to ML1 register.
This command should **not** be in normal use. It could be used to send any required bit sequence, replacing any other ML1 command.

FGM Sequence

```
send_ml1_word(data)
{
  send ZEF1MLTS,(data)
}

Equivalent sequence with 8-bit parameter list:
(data_lsb, data_msb)
{
  WORD data
  data=(data_msb << 8) | data_lsb
  send ZEF1MLTS,(data)
}
```

set_iel_speeds

set_iel_speeds(speed1, speed2)

Parameters

BYTE speed1 data rate from IEL 1
BYTE speed2 data rate from IEL 2

Parameter range limits

0 <= speed1 <= 1
0 <= speed2 <= 1

Action

Set the data rate from the IELs on both interfaces. Speed values of one correspond to fast (16KHz), values of zero correspond to slow (1KHz).

FGM Sequence

```
set_speeds (speed1, speed2)
{
  send ZEF2I1FN,(speed1)
  send ZEF2I2FN,(speed2)
}
```

set_test

set_test(select)

Parameters

BYTE select test number

Parameter range limits

0 <= select <=3

Action

Select the DPU test.

FGM Sequence

```
set_test(select)
{
  send ZEF2TSTS,(select)
}
```

set_sensor_cals

```
set_sensor_cals(pcal, scal)
Parameters
BYTE    pcal           primary sensor CAL status
BYTE    scal           secondary sensor CAL status

Parameter range limits
0 <= pcal <= 1
0 <= scal <= 1

Action
Set the CAL signal level on both the primary and secondary sensors.
FGM Sequence
set_sensor_cals(pcal, scal)
{
  send ZEF2PCLN,(pcal)
  send ZEF2SCLN,(scal)
}
```

set_sensor_flips

```
set_sensor_flips(pflip, sflip)
Parameters
BYTE    pflip          primary sensor FLIP status
BYTE    sflip          secondary sensor FLIP status

Parameter range limits
0 <= pflip <= 1
0 <= sflip <= 1

Action
Set the sensor FLIP signal levels on both sensors.
FGM Sequence
set_sensor_flips (pflip, sflip)
{
  send ZEF2PFLN,(pflip)
  send ZEF2SFLN,(sflip)
}
```

set_telemetry_option

```
set_telemetry_option(option)
Parameters
BYTE           option         telemetry mode
Parameter range limits
option = 0x2,0x3,0x4,0xa,0xb,0xc,0xd,0xf
Action
Set the instrument telemetry mode.
FGM Sequence
set_telemetry_option(option)
{
  send ZEF2TMMS,(option)
}
```

set_cal_mode

```
set_cal_mode(mode)
Parameters
BYTE    mode           cal mode number
Parameter range limits
0 <= mode <= 3
Action
Set the calibration mode.
FGM Sequence
set_cal_mode(mode)
{
  send ZEF2CLBS,(mode)
}
```

send_code_patch

send_code_patch()

Parameters

none

Parameter range limits

N/A

Action

Send_code_patch is a special command. Code patch command sequences must be agreed by the FGM team before transmission.

set_memory_dump

set_memory_dump()

Parameters

none

Parameter range limits

N/A

Action

Set the processor to dump the contents of memory defined in the parameter table into the science telemetry.

FGM Sequence

```
set_memory_dump()
{
  send ZEF2DMPS
}
```

enable_seu_monitor

enable_seu_monitor()

Parameters

void

Parameter range limits

N/A

Action

Set the seu monitor on.

FGM Sequence

```
enable_seu_monitor()
{
  send ZEF2SEUN,1
}
```

disable_seu_monitor

disable_seu_monitor()

Parameters

void

Parameter range limits

N/A

Action

Set the seu monitor off.

FGM Sequence

```
disable_seu_monitor()
{
  send ZEF2SEUN,0
}
```

enable_filtering

enable_filtering()

Parameters

void

Parameter range limits

N/A

Action

Set the filtering on.

FGM Sequence

```
enable_filtering()
{
  send ZEF2FILN,1
}
```

disable_filtering

```
disable_filtering()
Parameters
void
Parameter range limits
N/A
Action
Set the filtering off.
FGM Sequence
disable_filtering()
{
  send ZEF2FILN,0
}
```

select_primary_sensor

```
select_primary_sensor(sensor)
Parameters
WORD    sensor         select inboard or outboard as primary sensor
Parameter range limits
0 <= sensor <= 1
Action
Set the primary sensor to OB/IB. The secondary sensor is forced to be the other
sensor. Data value one is OB, zero is IB. Both sensors are set to auto ranging
on exit.
FGM Sequence
select_primary_sensor(sensor)
{
  send ZEF2PRNS,0x7
  send ZEF2SRNS,0x7
  send ZEF2POBS,(sensor)
  send ZEF2SOBS,(sensor ^ 0x01)
  send ZEF2PRNS,0x0
  send ZEF2SRNS,0x0
}

Equivalent sequence with 8-bit parameter list:
(sensor , not_sensor)
{
  send ZEF2PRNS,0x7
  send ZEF2SRNS,0x7
  send ZEF2POBS,(sensor)
  send ZEF2SOBS,(not_sensor)
  send ZEF2PRNS,0x0
  send ZEF2SRNS,0x0
}
```

configure_interface

```
configure_interfaces(data)
Parameters
WORD           data           16 bit interface configuration word
Parameter range limits
0x0 <= data <= 0xffff
Action
Set the configuration of the S/C interfaces.
FGM Sequence
configure_interfaces(data)
{
  send ZEF2CIMS,(data & 0xff00) >> 8
  send ZEF2CILS,(data & 0x00ff)
}

Equivalent sequence with 8-bit parameter list:
(d_lsb , d_msb)
{
  send ZEF2CIMS,(d_msb)
  send ZEF2CILS,(d_lsb)
}
```

configure_adcs

```
configure_adcs(data)
Parameters
WORD     data         16 bit ADC configuration word

Parameter range limits
0 <= data <= 0xffff

Action
Set the configuration of the FGM ADCs.

FGM Sequence
configure_adcs (data)
{
  send ZEF2CAMS,(data & 0xff00) >> 8
  send ZEF2CALS,(data & 0x00ff)
}

Equivalent sequence with 8-bit parameter list:
(d_lsb , d_msb)
{
  send ZEF2CAMS,(d_msb)
  send ZEF2CALS,(d_lsb)
}
```

set_parameter_base_address

```
set_parameter_base_address(base)
Parameters
BYTE     base         base address for subsequent parameter table writes

Parameter range limits
0 <= base <= 0xff

Action
Set the base address for parameter table writes.

FGM Sequence
set_parameter_base_address(base)
{
  send ZEF2PBAS,(base)
}
```

set_parameter_byte

```
set_parameter_byte(data)
Parameters
BYTE    data           parameter table update byte
Parameter range limits
0 <= data <= 0xff
Action
Uplink the next byte in the parameter table.
FGM Sequence
set_parameter_byte(data)
{
  send ZEF2PBYS,(data)
}
```

dpu_1_power_on

```
dpu_1_power_on()
Parameters
void
Parameter range limits
N/A
Action
Power ON FGM DPU 1
FGM Sequence
dpu_1_power_on()
{
  send ZEF1DP1N
}
```

dpu_2_power_on

dpu_2_power_on()
Parameters
void
Parameter range limits
N/A
Action
Power ON FGM DPU 2
FGM Sequence

```
dpu_2_power_on()
{
  send ZEF1DP2N
}
```

dpu_1_power_off

dpu_1_power_off()
Parameters
void
Parameter range limits
N/A
Action
Power OFF FGM DPU 1
FGM Sequence

```
dpu_1_power_off()
{
  send ZEF1DP1F
}
```

dpu_2_power_off

dpu_2_power_off()
Parameters
void
Parameter range limits
N/A
Action
Power OFF FGM DPU 2
FGM Sequence

```
dpu_2_power_off()
{
  send ZEF1DP2F
}
```

enable_dpu_reset

enable_dpu_reset()
Parameters
void
Parameter range limits
N/A
Action
Enable DPU reset control on active interface.
FGM Sequence

```
enable_dpu_reset()
{
  send ZEF1RESE
}
```

disable_dpu_reset

disable_dpu_reset()

Parameters

void

Parameter range limits

N/A

Action

Disable DPU reset control on active interface.

FGM Sequence

```
disable_dpu_reset()
{
  send ZEF1RESD
}
```

dpu_1_reset_on

dpu_1_reset_on()

Parameters

void

Parameter range limits

N/A

Action

Assert DPU 1 reset high.

FGM Sequence

```
dpu_1_reset_on()
{
  send ZEF1D1RN
}
```

dpu_1_reset_off

dpu_1_reset_off()

Parameters

void

Parameter range limits

N/A

Action

Assert DPU 1 reset low.

FGM Sequence

```
dpu_1_reset_off()
{
  send ZEF1D1RF
}
```

dpu_2_reset_on

dpu_2_reset_on()
Parameters
void
Parameter range limits
N/A
Action
Assert DPU 2 reset high.
FGM Sequence

```
dpu_2_reset_on()
{
  send ZEF1D2RN
}
```

dpu_2_reset_off

dpu_2_reset_off()
Parameters
void
Parameter range limits
N/A
Action
Assert DPU 2 reset low.
FGM Sequence

```
dpu_2_reset_off()
{
  send ZEF1D2RF
}
```

adc_1_power_on

adc_1_power_on()
Parameters
void
Parameter range limits
N/A
Action
Power ON FGM ADC 1.
FGM Sequence

```
adc_1_power_on()
{
  send ZEF1AD1N
}
```

adc_2_power_on

adc_2_power_on()
Parameters
void
Parameter range limits
N/A
Action
Power ON FGM ADC 2.
FGM Sequence

```
adc_2_power_on()
{
  send ZEF1AD2N
}
```

adc_1_power_off

adc_1_power_off()
Parameters
void
Parameter range limits
N/A
Action
Power OFF FGM ADC 1.
FGM Sequence

```
adc_1_power_off()
{
  send ZEF1AD1F
}
```

adc_2_power_off

adc_2_power_off()
Parameters
void
Parameter range limits
N/A
Action
Power OFF FGM ADC 2.
FGM Sequence

```
adc_2_power_off()
{
  send ZEF1AD2F
}
```

msa_power_on

msa_power_on()
Parameters
void
Parameter range limits
N/A
Action
Power ON FGM MSA.
FGM Sequence

```
msa_power_on()
{
  send ZEF1MSAN
}
```

msa_power_off

msa_power_off()
Parameters
void
Parameter range limits
N/A
Action
Power OFF FGM MSA.
FGM Sequence

```
msa_power_off()
{
  send ZEF1MSAF
}
```

enable_outboard_sensor

enable_outboard_sensor()
Parameters
void
Parameter range limits
N/A
Action
Enable control of outboard sensor on active interface.
FGM Sequence

```
enable_outboard_sensor()
{
  send ZEF1OBSE
}
```

enable_inboard_sensor

enable_inboard_sensor()
Parameters
void
Parameter range limits
N/A
Action
Enable control of inboard sensor on active interface.
FGM Sequence

```
enable_inboard_sensor()
{
  send ZEF1IBSE
}
```

disable_outboard_sensor

disable_outboard_sensor()
Parameters
void
Parameter range limits
N/A
Action
Disable control of outboard sensor on active interface.
FGM Sequence

```
disable_outboard_sensor()
{
  send ZEF1OBSD
}
```

disable_inboard_sensor

disable_inboard_sensor()
Parameters
void
Parameter range limits
N/A
Action
Disable control of inboard sensor on active interface.
FGM Sequence

```
disable_inboard_sensor()
{
  send ZEF1IBSD
}
```

enable_iel

enable_iel()
Parameters
void
Parameter range limits
N/A
Action
Enable IEL on the active interface.
FGM Sequence

```
enable_iel()
{
  send ZEF1IELE
}
```

disable_iel

disable_iel()
Parameters
void
Parameter range limits
N/A
Action
Disable IEL on the active interface.
FGM Sequence

```
disable_iel()
{
  send ZEF1IELD
}
```

clear_ml2_stack

clear_ml2_stack()

Parameters

void

Parameter range limits

N/A

Action

Clear contents of ML2 stack on active interface.

FGM Sequence

```
clear_ml2_stack()
{
  send ZEF1MSCS
}
```

interface_power_on

interface_power_on()

Parameters

void

Parameter range limits

N/A

Action

Power ON FGM inactive interface.

FGM Sequence

```
interface_power_on()
{
  send ZEF1INTN
}
```

interface_power_off

interface_power_off()

Parameters

void

Parameter range limits

N/A

Action

Power OFF FGM inactive interface.

FGM Sequence

```
interface_power_off()
{
  send ZEF1INTF
}
```

5. JSOC/ESOC command sequence aliases

PI_ALIAS/JSOC name	ESOC_NAME	ESOC_DESCRIPTION	COMMENT
FGM_OFF_OPM1	SFGMJ000	E_FGM000	off_to_FGMOPM1
FGM_OPM1_OFF	SFGMJ001	E_FGM001	FGMOPM1_to_off
FGM_OPM1_OPM2	SFGMJ002	E_FGM002	FGMOPM1_to_FGMOPM2
FGM_OPM1_OPM3	SFGMJ003	E_FGM003	FGMOPM1_to_FGMOPM3
FGM_OPM1_OPM4	SFGMJ004	E_FGM004	FGMOPM1_to_FGMOPM4
FGM_OPM1_OPM5	SFGMJ005	E_FGM005	FGMOPM1_to_FGMOPM5
FGM_OPM1_OPM7	SFGMJ006	E_FGM006	FGMOPM1_to_FGMOPM7
FGM_OPM1_OPM8	SFGMJ007	E_FGM007	FGMOPM1_to_FGMOPM8
FGM_OPM2_OPM1	SFGMJ008	E_FGM008	FGMOPM2_to_FGMOPM1
FGM_OPM2_OPM3	SFGMJ009	E_FGM009	FGMOPM2_to_FGMOPM3
FGM_OPM3_OPM1	SFGMJ00A	E_FGM00A	FGMOPM3_to_FGMOPM1
FGM_OPM3_OPM2	SFGMJ00B	E_FGM00B	FGMOPM3_to_FGMOPM2
FGM_OPM4_OPM1	SFGMJ00C	E_FGM00C	FGMOPM4_to_FGMOPM1
FGM_OPM5_OPM1	SFGMJ00D	E_FGM00D	FGMOPM5_to_FGMOPM1
FGM_OPM6_OPM1	SFGMJ00E	E_FGM00E	FGMOPM6_to_FGMOPM1
FGM_OPM7_OPM1	SFGMJ00F	E_FGM00F	FGMOPM7_to_FGMOPM1
FGM_OPM8_OPM1	SFGMJ010	E_FGM010	FGMOPM8_to_FGMOPM1
FGM_CAL	SFGMJ011	E_FGM011	FGMCAL
ENABLE_EVENT_REC	SFGMJ016	E_FGM016	enable_event_recognition
DISABLE_EVENT_REC	SFGMJ017	E_FGM017	disable_event_recognition
MSA_TRIGGER	SFGMJ018	E_FGM018	set_msa_trigger
MSA_UNTRIGGER	SFGMJ019	E_FGM019	set_msa_untrigger
DUMP_PARAM_TABLE	SFGMJ01C	E_FGM01C	dump_parameter
KEYHOLE_ADDRESS	SFGMJ01D	E_FGM01D	set_keyhole_address
MEMO_DUMP_PARAMS	SFGMJ01E	E_FGM01E	set-memory_dump
AVERAGING_LENGTH	SFGMJ01F	E_FGM01F	set_averaging_length
VARIANCE_THRES	SFGMJ020	E_FGM020	set_variance_threshold
PI_ALIAS/JSOC name	**ESOC_NAME**	**ESOC_DESCRIPTION**	**COMMENT**

VARIANCE_OFFSET	SFGMJ021	E_FGM021	set_variance_offset
BX_THRESHOLD	SFGMJ022	E_FGM022	set_bx_threshold
BX_OFFSET	SFGMJ023	E_FGM023	set_bx_offset
B2_THRESHOLD	SFGMJ024	E_FGM024	set_b2_threshold
B2_OFFSET	SFGMJ025	E_FGM025	set_b2_offset
BL2_THRESHOLD	SFGMJ026	E_FGM026	set_bl2_threshold
BL2_OFFSET	SFGM027	E_FGM027	set_bl2_offset
ZY2_THRESHOLD	SFGM028	E_FGM028	set_zy2_threshold
ZY2_OFFSET	SFGM029	E_FGM029	set_zy2_offset
MSA_DECIMATION	SFGM02A	E_FGM02A	set_msa_decimation
DETECTION_SWITCH	SFGM02B	E_FGM02B	set_detection_switches
MODESWITCH_TIME	SFGM02C	E_FGM02C	set_modeswitch_times
IB_OFFSET	SFGM02D	E_FGM02D	set_ib_offsets
OB_OFFSET	SFGM02E	E_FGM02E	set_ob_offsets
PRIMARY_RANGE	SFGM02F	E_FGM02F	set_primary_range
SECONDARY_RANGE	SFGM030	E_FGM030	set_secondary_range
FGM_EXT_OPM1	SFGM050	-	FGMEXT_to_FGMOPM1
FGM_OPM1_EXT	SFGM059	-	FGMOPM1_to_FGMEXT